Lessons Learned

Lessons Learned — Accumulated Knowledge

2026-04-04: P0 Endpoint Hallucination — LightRAG /upload vs /documents/text

Problem: Builder agent hallucinated `/upload` endpoint for LightRAG when correct endpoint is `/documents/text`. Error passed through entire system without detection — code written, deployed, and failed in production demo.

Root Cause Analysis:

1. Agent assumed endpoint name based on "sounds right" pattern matching
2. No endpoint verification hook in place
3. qa-19 quality gate lacked endpoint testing
4. Tool-registry.db was inactive — no nightly endpoint audit

Impact: Demo-blocking bug in LumisCare, revealed systemic vulnerability to API hallucinations.

Solution (3-Part):

1. **P1: Anti-Hallucination Hook** — hallucination-detector.py now has KNOWN_API_ENDPOINTS dict + check_phantom_endpoints()
   - Blocks Write/Edit with known invalid endpoints
   - Examples: `/upload` → use `/documents/text`
2. **P2: Nightly Audit Daemon** — tool-sync-audit.js scans all tools for stale endpoints
   - Tests each HTTP endpoint via HEAD (timeout 3s)
   - Logs to health-events.db
   - Alerts Slack if stale endpoints found
   - LaunchAgent: com.john.tool-sync-audit (03:00 daily)
3. **P3: Quality Gate Check** — qa-19.js now includes Check #20: Endpoint Verification

- Parses GOTCHA for HTTP endpoints
- Tests each before task completion
- Blocks mc.js done if endpoints fail

Pravilo (Rule 10 — agent-anti-hallucination.md):

```
Before using any HTTP endpoint:
1. curl -s http://localhost:PORT/health
2. Check OpenAPI spec: curl -s http://localhost:PORT/openapi.json
3. Verify in KNOWN_API_ENDPOINTS (hallucination-detector.py)
NEVER assume endpoint exists because it "sounds right"
```

Prevention for future:

- hallucination-detector.py checks all Write/Edit for phantom endpoints
- tool-sync-audit.js catches stale endpoints weekly
- qa-19 Check #20 blocks tasks with dead endpoints
- enforcement.json enforces endpoint_check blocking

Lekcija: API hallucinations are deterministic errors — agent + endpoint name that sounds right = confident wrong code. Solution is three-layer: hook prevention + nightly audit + quality gate. Builder agent can't self-verify, so verification must be external + automated.

2026-02-12: NIKAD BUILD od self-generated spec-a bez CEO approval

Problem: John je na DROP projektu sam napravio UI/UX spec (competitor analysis, 3 dizajn opcije), pa odmah krenuo graditi full app — 97 fajlova, 24K LOC. Bez ijednog Alemovog odobrenja na spec. Rezultat: kod zaglavljen na wrong git branch, prazan `drop-app/` dir, wasted tokens, Alem ne zna šta je napravljeno.

Root Cause: Nedostajao approval gate između faze Research/Spec i faze Build. John je tretirao self-generated spec kao odobren spec.

Pravilo (ZAKON):

1. **Research** → OK, radi slobodno
2. **Spec/Proposal draft** → OK, radi slobodno
3. **BUILD** → STOP. Explicit CEO odobrenje na spec PRIJE prvog LOC.
4. Ako CEO nije pregledao spec, spec NE POSTOJI kao basis za build.
5. Self-generated spec ≠ Approved spec. NIKAD.

Recovery: `fontelepay/` auto-backup branch merged to master. Kod recovered.

Fix nivo: Rule (ovaj fajl) + HiveMind (#76) + MEMORY.md. Idealan fix bi bio hook koji blokira build bez approved spec — ali approval gate je human decision, teško za hook.

Lekcija: AI može napraviti spec, ali samo čovjek može ODOBRITI spec. Bez odobrenja, build je gubitak resursa.

2026-02-08: Next Steps MORAJU postati MC taskovi

Problem: Session log imao "Next Steps" ali nikad nisam kreirao MC taskove za njih. Rezultat: 2 akcije (Edita MC onboarding + Mini SSH update) izgubljene jer niko ne čita session log automatski.

Root Cause: Session-save workflow zapisuje next steps u markdown ali nema korak koji ih pretvara u MC taskove.

Fix: PRAVILO — prije kraja sesije, svaki "Next Step" iz session state-a MORA postati `mc.js add` task. Session state je za kontekst, MC je za akciju. Ako nije u MC-u, ne postoji.

Lekcija: Passive documentation (markdown) ≠ active tracking (MC). Ako nešto treba biti urađeno, mora biti task.

2026-02-04: Task Management + Problem Solving Enforcement

Problem: Skip-ovao sam task tracking i problem solving proces, delegirao agenta bez proper requirements gathering, agent riješio pogrešan problem.

Root Cause Analysis:

1. Nisam dodao task u tasks.db
2. Nisam pratio problem-solving.md proces (koraci 1-6)
3. Spawn-ovao agenta sa PRVIM rješenjem (email infrastructure umjesto client communication system)
4. Agent radio PLAN fazu solo - trebalo John + client
5. Nisam završio Next Steps iz SESSION-STATE

Impact: Alem dobio pogrešno rješenje, izgubljeno vrijeme, "veći problem" kreiran

Solution Implemented:

1. ✅ Kreiran ~/system/tools/start-task.sh - mandatory validation script
2. ✅ Update MEMORY.md sa CORE PROTOCOL sekcijom
3. ✅ Dokumentovano u lessons-learned.md (ovdje)
4. ✅ boot.sh reminder dodan

Validation:

- start-task.sh blokira izvršenje ako nisu zadovoljeni koraci 1-4
- Checklist forsira: task.db entry → problem solving (1-6) → company delegation check
- MEMORY.md učitava se na session start sa reminder-om

Prevention:

- NIKAD ne radim ništa bez `bash ~/system/tools/start-task.sh` prvo
- Script deterministic - ne mogu skip-ovati
- boot.sh prikazuje reminder na session start

Key Mantras:

- "Prvo rješenje" ≠ "Najbolje rješenje"
- Research PRIJE implementacije (WebSearch 2+ izvora)
- 2-3 opcije UVIJEK, ne samo jedna
- PLAN phase = John + client, ne agent solo

2026-02-12: Sub-agent Validator Hallucination — "PASS" na pogrešan format

Problem: John mijenjao Claude Code hooks format u `.claude/settings.json`. Napisao `matcher: {}` (objekt) umjesto `matcher: "*"` (regex string). Pozvao haiku sub-agenta kao "testera" — agent rekao PASS. Alem pokrenuo Claude, dobio isti error.

Root Cause (2 nivoa):

1. **John nije pročitao dokumentaciju** prije izmjene config formata. Pretpostavio format iz error poruke.
2. **Sub-agent validirao John-ov output** umjesto da nezavisno provjeri spec. Haiku agent nema znanje o novom hooks formatu — hallucinate-ovao da je ispravan.

Impact: Alem dobio error 2x, izgubljeno povjerenje u "tester" agente.

Fix:

1. **Pravilo:** NIKAD mijenjaj config/schema format bez čitanja oficijelne dokumentacije (WebFetch/WebSearch)
2. **Pravilo:** Validator/tester sub-agent MORA imati instrukciju da NEZAVISNO provjeri source of truth (docs URL, spec file), NE da validira caller-ov rad
3. **Anti-pattern:** "Provjeri da sam dobro uradio" ≠ testiranje. Testiranje = nezavisna verifikacija protiv spec-a.

Key Mantras:

- Docs first, code second
- Validator ≠ rubber stamp
- Haiku ne zna ono što ne zna — ne koristi ga za format verifikaciju bez docs referenci

---

2026-02-16: UI promjene bez prethodne provjere dizajn referenci (Drop #979)

Problem: Landing page imao "Virtuelt kort" feature koji je kontradiktoran Drop PSD2 pass-through modelu (no cards, no wallet). Kad sam to fixovao u "Kontooversikt", napravio sam promjenu BEZ prethodne provjere Make exporta. Alem morao eksplicitno reći: "Jeli li validirao imas vizuelno u MAKE pa tako treba da je i UI."

Root Cause: Dva propusta:

1. **Niko nije validirao original** — "Virtuelt kort" je ušao u kod bez provjere protiv Make dizajna koji NEMA Cards screen
2. **Fix bez referenci** — Ja sam fixovao sadržaj iz glave umjesto da prvo pročitam Make export i repliciram TAČNO šta je tamo

Impact: Srećom output je bio tačan (Make JESTE imao BankAccounts, ne Cards), ali proces je bio pogrešan. Da je Make imao nešto drugačije, ja bih opet deployao pogrešno.

Fix:

1. **Drop CLAUDE.md** — Dodan "UI Source of Truth" sekcija sa Make export putanjom i pravilom "BEFORE any UI change, read Make component"
2. **visual-verification.md** — Dodan korak 0: "REFERENCA PRIJE KODA" — zabranjeno mijenjati UI pa tek onda provjeriti dizajn
3. **HiveMind** — Logirano za budući kontekst

Lekcija: Redoslijed je uvijek: dizajn → kod → verifikacija. Nikad: kod → (možda) verifikacija.

2026-02-16: UVIJEK koristi official brand template za firmine dokumente

Problem: Kreirao PDF za SpareBank 1 pitch i poslao Alemu. Prvo poslao markdown umjesto PDF-a. Onda napravio PDF sa pogrešnim bojama (#0B6E35 Drop green umjesto #00E5A0 ALAI green), pogrešnim cover dizajnom (light umjesto dark navy), bez korištenja official template-a. Alem: "Gdje si nasao ovaj template u ALAI? TO nije pravi."

Root Cause: Nema pravilo koje forsira provjeru brand guidelines i template-a PRIJE kreiranja bilo kakvog firmino-brendiranog dokumenta. John je improvizirao dizajn umjesto da pročita brand-guidelines.md i pogleda template slike.

Pravilo (ZAKON):

1. **SVAKI dokument sa ALAI branding** mora PRVO pročitati `~/ALAI/brand/brand-guidelines.md`
2. **SVAKI dokument** mora koristiti official boje: Primary Green #00E5A0, Dark Navy #0F172A
3. **SVAKI PDF** mora vizualno odgovarati template-ima iz `~/ALAI/brand/templates/` (presentation.png za prezentacije, letter.png za pisma, invoice.png za fakture)
4. **NIKAD ne improvizuj brand** — ako ne znaš kako izgleda, PROČITAJ template prije nego počneš
5. **GOTCHA C (Context) sekcija** za branded dokumente MORA sadržavati "brand-guidelines.md read" i navesti tačne boje

Brand Quick Reference:

- Primary Green: `#00E5A0` (NE #0B6E35 — to je Drop green)
- Dark Navy: `#0F172A` (cover background)
- Bright Green: `#22C55E` (accent)
- Font: Inter (Regular, Medium, SemiBold, Bold)
- Logo: `~/ALAI/brand/alai-logo-primary.png`
- Templates: `~/ALAI/brand/templates/`
- Footer: "ALAI Holding AS · Org.nr 932 516 136 · info@alai.no · alai.no"

Fix nivo: Rule (ovaj fajl) + HiveMind + MEMORY.md

Lekcija: Branded dokument bez brand guidelines = amaterski. Uvijek čitaj guidelines PRIJE dizajna, nikad poslije.

2026-02-16: Agent .md hooks: sekcija OVERRIDUJE globalne hookove

Problem: Builder agent za task #1039 napisao kod bez GOTCHA checkliste. Validator potvrdio: `/tmp/gotcha-task-1039.md` — NOT FOUND. gotcha-enforcer.py nikad nije blokirao jer se nikad nije pokrenuo.

Root Cause: Agent `.md` fajlovi (builder.md, frontend-builder.md, backend-builder.md, design-builder.md) imali `hooks:` sekciju u YAML frontmatteru. Kad agent definira hooks — to ZAMIJENI globalne hookove iz settings.json, NE merge-uje ih. Rezultat: SVE PreToolUse enforcement hookove (gotcha-enforcer, plan-enforcer, security-guard, hallucination-detector, pii-scanner) su zaobiđeni.

Impact: 4 agenta radila bez ikakvog enforcement-a. Ironično, design-validator (jedini hook u agent .md) je VEĆ bio registrovan globalno u settings.json — lokalne kopije su bile duplikati koji su samo blokirali ostale hookove.

Fix:

1. Uklonjene `hooks:` sekcije iz sva 4 agenta (builder, frontend-builder, backend-builder, design-builder)
2. Svi agenti sada nasljeđuju SVE globalne hookove iz settings.json
3. design-validator ostaje u globalnom PostToolUse (settings.json linija 142-147)
4. Backup: `~/system/backups/setup-changelog/20260216-184634/`

Pravilo (ZAKON):

- **NIKAD ne dodavaj `hooks:` sekciju u agent .md fajlove** — uvijek koristi globalni settings.json za hookove
- Ako agent treba specifičan hook — dodaj ga u globalni settings.json sa odgovarajućim matcher-om
- Agent .md definira samo: name, model, tools — NIKAD hooks

Fix nivo: Deterministic (uklanjanje hooks: iz agent .md) + Rule (ovaj fajl) + HiveMind (#7191) + CHANGELOG

Vercel Deployment

- **NE koristi stare `builds` + `routes`** u vercel.json — koristi moderni pristup:

```
{ "outputDirectory": "public" }
```

- **API folder** `/api` se automatski detektuje — ne treba build config
- **Environment variables** moraju biti na PRAVOM projektu — provjeri `vercel env ls`

Resend Email

- **Custom domena** zahtijeva DNS verifikaciju u Resend dashboardu
- **API key** mora biti na istom Vercel projektu gdje je API endpoint
- **Testiranje:** 404 = deploy config problem. 500 = API key/domena problem.

Telegram Bot Auth

- **NEVER use direct API key for Telegram bot** — use Claude CLI spawn (OAuth)
- API keys run out of credits, OAuth doesn't
- Always verify auth method when implementing bot changes
- Bot file: `~/system/comms/telegram-claude-bridge.js`
- LaunchAgent: `~/Library/LaunchAgents/com.john.telegram-bot.plist`

General

- Verify tool output format before chaining into another tool
- Don't assume APIs support batch operations — check first
- When a workflow fails mid-execution, preserve intermediate outputs before retrying
- **Provjeri pravi projekt** prije dodavanja env vars
- **Test endpoint** nakon svakog deploya

Background Agents & Security Hooks

- **Background agenti (`run_in_background: true`) nemaju write permissije** — security hook blokira Write, Edit i Bash
- Koristi background agente SAMO za research, audit, čitanje — nikad za pisanje fajlova
- Ako background agent treba nešto napisati, vrati rezultat u glavnu sesiju i piši odatle
- Naučeno: EVApp background agent nije mogao kreirati fajlove jer je hook blokirao — morali smo ručno iz glavne sesije

Testing

- "HTML exists" ≠ "It works"
- grep/curl is NOT a visual test

- Automatski testovi su supplement, NE zamjena za vizuelni QA

2026-02-04: Problem-Solving Enforcement System

Problem: John preskakao CORE PROTOCOL - išao direktno na implementaciju bez analize.

Root cause: Validation flag bio statičan, nikad se nije resetovao.

Rješenje implementirano:

1. `boot.sh` briše `/tmp/claude-task-validated` na početku sesije
2. `security-guard.py` traži problem-solving dokumentaciju u `/tmp/claude-problem-solving.md`
3. Dokumentacija mora imati 5 sekcija: PROBLEM, RESEARCH, OPCIJE, EVALUACIJA, ODLUKA
4. Bootstrap exception: Write dozvoljeno SAMO na problem-solving fajl
5. Kad dokumentacija kompletna → auto-validacija → flag kreiran

Workflow:

- Nova sesija → flag resetovan → blokirani Write/Edit/Bash
- Ja dokumentiram proces → hook provjerava → auto-validates
- Tek onda mogu implementirati

Fajlovi izmijenjeni:

- `~/system/boot.sh` - dodano brisanje flaga
- `~/.claude/hooks/security-guard.py` - dodana problem-solving validacija

Lekcija: Enforcement mora biti automatski i neizbježan. Ako se može preskočiti, bit će preskočen.

2026-02-04: Hooks Can Only Approve/Block, NOT Modify

Problem: Agent-protocol-enforcer.py vraćao `updatedInput` misleći da će Claude Code koristiti modificirani prompt. Agenti su i dalje pitali tehnicka pitanja.

Root Cause: `updatedInput` nije podržan u Claude Code hooks API. Hooks mogu samo:

- `exit 0` → approve (allow tool call)

- `exit 2` → block (reject tool call with stderr message)

Hooks su GATE kontrola, ne transformacija.

Fix:

1. Hook sada BLOKIRA Task bez CORE PROTOCOL markera
2. John mora eksplicitno dodati protokol u svaki agent prompt
3. Built-in tipovi (Explore, Plan, Bash) su izuzeti - imaju svoje instrukcije

Fajl: `~/.claude/hooks/agent-protocol-enforcer.py`

Lekcija: Ne pretpostavljaj da feature postoji. Testiraj da hook STVARNO radi kako misliš.

2026-02-04: DocuSeal — Paid Only

Problem: Koristili DocuSeal za digitalni potpis NDA/ugovora sa Wizard NUF-om. Nije radilo.

Root Cause: DocuSeal nema free plan - zahtijeva plaćenu pretplatu za production use.

Impact: Wizard NUF onboarding ostao bez potpisanih dokumenata. Pipeline testiran ali faza 3 (NDA) i 5 (Contract) nisu kompletne.

Next: Task #52 - naći alternativu za digitalni potpis koja ima free tier ili je self-hosted.

Lekcija: Prije integracije sa SaaS alatom, provjeri pricing i limits. "Free trial" ≠ "Free tier".

2026-02-17: Presko?en /hop-build pipeline — output ne valja (Drop #1309)

Problem: Task #1309 (Drop mobile production build) — John je preskočio /hop-build pipeline. Umjesto toga: ručno spawnao 3 builder agenta paralelno, napisao surface-level GOTCHA checklist samo da prođe hook, nije koristio validator agente. Rezultat: Alem dobio APK koji "ne valja". ZAKON #0 prekršen OPET.

Root Cause (iz analize):

1. **Nema enforcement za /hop-build** — gotcha-enforcer provjerava GOTCHA checklist ali NE provjerava da li je hop-build PROCES korišten

2. **Skill invocation je dobrovoljna** — nema hook koji detektuje "trebao si koristiti /hop-build ali nisi"
3. **Builder spawn bez process state** — orchestrator-delegation-enforcer dozvoljava direktan builder spawn, ne razlikuje "via hop-build" od "ručno"
4. **MEDIUM priority nema plan enforcer** — plan-enforcer.py zahtijeva plan JSON samo za HIGH priority

Impact: 3 builder agenta radila bez proper plana, bez validatora, bez verifikacijske faze. Output deployovan na Expo bez validacije. Alem eksplicitno rekao: "ovo sto si mi dao ne valja" i "kreni ispočetka".

Fix (tiered):

1. **Hook (WARNING):** gotcha-enforcer.py CHECK 5 — warn kad MEDIUM+ task nema `/tmp/hop-build-started-{id}` marker
2. **Skill update:** /hop-build Phase 1 sad kreira marker fajl
3. **ZAKON #5:** "Svaki implementation task MORA koristiti /hop-build" (MEMORY.md)
4. **Lessons-learned:** Ovaj zapis

Zašto WARNING a ne BLOCK: Novo pravilo — treba validacijski period. Ako se pokaže da false positive rate je nizak, escalirat će se na exit 2 (BLOCK).

Lekcija: GOTCHA checklist je "razmisli prije kodiranja". /hop-build je "slijedi PROCES kodiranja". Jedno bez drugog = half-assed. Task #1309 dokazuje: razmišljanje bez procesa → shortcuti → broken output.

2026-02-04: Agenti moraju znati za sistem

Problem: Agenti kad zapnu pitaju umjesto da koriste problem-solving proces.

Root Cause: Agentima nisam davao informaciju O sistemu — samo task. Ne znaju da `/tmp/claude-problem-solving.md` postoji.

Fix: Kreiran `~/system/agents/BOOTSTRAP.md` — svaki agent prompt počinje sa "Pročitaj BOOTSTRAP.md".

Lekcija: Agent bez konteksta o sistemu će raditi ad-hoc. Mora znati KAKO rješavamo probleme, ne samo ŠTA treba uraditi.

Lesson Learned: PI Orchestrator Task Routing Failures

Date: 2026-03-11 **Context:** World-Class Gap Analysis — 13 parallel tasks **Impact:** 4+ hours delay, 3 rounds of manual re-dispatching

Root Causes Found

1. delegate_task ? Event Bus drops tasks silently

- Dispatched 13 tasks via delegate_task, only 4-6 arrived as MC tasks
- **No error returned** — delegate_task says "Event emitted" but no guarantee of delivery
- **Fix needed:** Event bus must ACK with MC task ID, or delegate_task must verify creation

2. Owner mismatch: delegate_task assigns to "pi-orchestrator" but orchestrator queries --owner john

- `pi-orchestrator.js` line 1087: `next-task --owner john`
- delegate_task creates tasks with owner = "pi-orchestrator"
- Result: tasks invisible to orchestrator
- **Fix needed:** Either delegate_task should set owner=john, OR orchestrator should query both owners

3. mc.js start puts tasks in "in_progress" — orchestrator only picks up "open"

- When manually starting tasks with `mc.js start`, status becomes "in_progress"
- `next-task` only returns "open" status tasks
- Result: manually started tasks never get picked up
- **Fix needed:** Orchestrator should also consider "in_progress" tasks that have no active worker, OR document that mc.js start should NOT be used for orchestrator-managed tasks

4. Classifier sends research tasks to human-queue (complexity=5)

- Gap analysis research tasks classified as complexity=5 → auto-routed to human-queue
- These are research/analysis, not architecture decisions — complexity=4 is appropriate
- **Fix needed:** Classifier prompt should distinguish "deep research" from "architecture decision requiring human"

5. Classifier sends tasks to qwen3:8b which fails on complex analysis

- Some tasks misclassified as complexity=1/devops → qwen3:8b on forge → fails
- **Fix needed:** Minimum complexity floor for H-priority tasks (never < 3)

Correct Workflow (Until Fixed)

1. Create tasks directly with `mc.js add "title" --priority H --owner john`
2. Do NOT use `mc.js start` — let orchestrator pick them up
3. Do NOT rely on delegate_task for batch dispatching — verify MC task creation
4. After delegate_task, always check `mc.js list --owner john --status open` to confirm

Systemic Fix Required

- [ ] Event bus delivery guarantee (at-least-once with ACK)
- [ ] Owner alignment: delegate_task → owner=john
- [ ] Classifier: H-priority → minimum complexity=3
- [ ] Classifier: "research/analysis" domain → never human-queue
- [ ] mc.js: add `reopen` command to reset in_progress → open

CI/CD & Production Monitoring (2026-03-12)

Incident: getdrop.no served drop-app instead of landing page for 7 days. No one noticed except CEO.

Root Cause

- AWS App Runner silently claimed getdrop.no as custom domain during a deploy session
- No automated check verifies "what content does our domain actually serve?"
- No uptime/content monitoring on any production URL
- CEO is the monitoring system — not scalable

Lessons

1. **Every production URL must have a smoke test** — not just health check, but CONTENT verification (expected title, expected response body)
2. **Domain ownership must be explicit and audited** — document which service owns which domain. Alert on any change.
3. **Deploy pipelines must verify the DESTINATION, not just the build** — ZAKON #10 says "verify on destination" but we only verify locally
4. **CI must GATE deploy** — deploy should require CI pass. Currently deploy is independent of CI.
5. **Infrastructure changes (DNS, custom domains, TF apply) must go through PR review** — never ad-hoc CLI commands
6. **One fix for ALL products, not per-product** — every fix must be systemic, applied to Drop AND Tok AND Bilko AND Lobby AND Plock AND BasicFakta

Required Actions (systemic, all products)

- ☐ Uptime monitoring for ALL production URLs (UptimeRobot/Checkly)
- ☐ Smoke test cron: verify content, not just HTTP 200
- ☐ Deploy gate: CI pass required before deploy
- ☐ Post-deploy verification: health + content + screenshot
- ☐ Domain audit: document service→domain mapping, alert on changes
- ☐ Terraform plan in PR (never ad-hoc apply)

2026-04-08: Testing Failure — Agents Write Tests That Cannot Fail (Drop)

Analysis by: Petter Graff + James Whittaker framework **Context:** 10+ consecutive CEO test failures. CEO found bugs in 5 minutes that 1232 E2E tests missed. **Full analysis:** `~/system/rules/lessons-learned-testing-2026-04-08.md`

Root Cause

Test agents design tests to PASS, not to FIND BUGS. This is a design philosophy failure, not a quantity failure.

Measured failures in Drop E2E suite:

- 48 instances of `test.skip()` — tests that lie about coverage
- 100+ instances of `.catch(() => false)` — failures silently swallowed
- 10 of 29 pages tested for basic load (19 pages never visited)
- 0% of tests use UI navigation (all use `page.goto()` direct URLs)
- All BankID tests mock the response — testing the mock, not the app
- Tests accept multiple outcomes: `expect([200, 404])` — 404 is accepted as "ok"

The 5 Behavioral Differences (CEO vs Agent)

1. CEO tests EXPECTATIONS. Agents test ASSERTIONS.
2. CEO tests JOURNEYS. Agents test COMPONENTS.
3. CEO tests WHAT EXISTS. Agents test WHAT THEY BUILT.
4. CEO tests CURRENT STATE (full regression). Agents test WHAT CHANGED.
5. CEO STOPS on ambiguity. Agents SKIP on ambiguity.

Anti-Patterns (BANNED in all E2E tests)

- `test.skip()` without a linked issue
- `.catch(() => false)` or `.catch(() => {})` in test assertions
- `expect([200, 404]).toContain(status)` — accepting failure as success
- `page.goto()` for navigation after initial load — must click UI elements
- `page.route()` mocking in E2E tests — test the real app
- Hardcoded page lists — must discover pages from `find src/app -name "page.tsx"`
- `.first()` on ambiguous locators — be specific

Required Patterns

- One expected outcome per assertion (not multiple acceptable)
- Click UI elements, don't goto URLs
- Test against deployed URL, not localhost
- Every page in the app must be visited (crawl-and-verify after each deploy)
- Financial data must be numerically verified (not string presence)
- Tests must FAIL when things break, never SKIP

Whittaker's 7 Tours (run after every deploy)

1. Guidebook Tour — follow the primary user path by clicking, not goto
2. Money Tour — verify every number on every screen (fee, rate, total, recipient)
3. Landmark Tour — navigate ONLY via visible UI elements
4. Intellectual Tour — test hardest features with complex inputs
5. FedEx Tour — follow data creation to completion and verify it matches
6. Garbage Collector Tour — visit ALL pages, including least-used ones
7. Bad Neighborhood Tour — re-test every previous CEO-found bug scenario

Solution Implemented

- James Whittaker agent: `~/system/agents/identities/james-whittaker.md`
- Drop E2E Whittaker tours: `/Users/makinja/ALAI/products/Drop/tests/e2e/whittaker-tours.spec.ts`
- 30 Scenario checklist: `/Users/makinja/ALAI/products/Drop/tests/DROP-30-SCENARIOS.md`
- Feedback rule: `~/.claude/projects/-Users-makinja/memory/feedback_testing_root_cause.md`

Lekcija: 1232 tests that skip on failure are worse than 10 tests that actually fail when things break. The required shift: tests must be designed to FIND bugs, not to PASS.

2026-06-12 — Generalizable process fixes (SnowIT-SEO OAuth session) — apply to ALL projects/clients

Memo: `~/.claude/projects/-Users-makinja/memory/feedback_generalizable_corrections_2026-06-12.md`.

- **B — Verify subagent claims by live outcome, not their word.** A subagent reported OAuth "FULLY ACTIVE" while prod had a silent credential mismatch (new client_id + old leftover secret). Never relay an agent's "done/works" to CEO without an independent live check. Put in every dispatch brief.
- **C — Verify external-platform assumptions at the vendor's own source before architecting.** A whole plan branch rested on a false "scopes are restricted -> $15-30k CASA" assumption; Google's own console showed Sensitive/non-sensitive. Confirm scope tiers / quotas / API existence / pricing from the vendor, not memory or a prior agent.
- **D — Cloud tenant isolation (now a ~/CLAUDE.md guardrail).** Per-client/product cloud resources go in that tenant's own project/account, never a shared default.
- **E — Validate the credential PAIR when changing an ID/secret.** Probe the provider token endpoint with a dummy code: invalid_grant = valid pair; invalid_client = mismatch. Do before declaring an integration live; include in deploy-brief evidence.

Revision #8
Created 2026-02-17 22:14:44 UTC by John
Updated 2026-07-19 20:00:15 UTC by John